Database & Knowledge Based Approaches to Information Retrieval; A Comparative Study

Dong-Hyung Kim(Bowling Green State Univ./MBA)
Srinvasan Raghunathan(Bowling Green State Univ./Ph.D)

ACKNOWLEDGEMENTS

I wish to acknowledge the people who inspired me to believe that a machine can have intellgence. They are the ones who envisioned the futue of mankind. I specially appreciate to my advisor Dr. Srinvasan raghunathan, who was always supportive and encouraged to finish this research. I also sincerely thank for my parents who are always concerned of my well-being. Without them, what I achived here cannot be possible.

I. Introduction

A computerized information system is a critical element for business organizations to be competitive in free market. Acquisition and retrieval of relevant information at the right time have become a basic requirement for the organizations to be successful in today's very competitive market. One of the core components of the overall information system in an organization is the database. The database management system is an effective and efficient way of organizing and managing data to create appropriate information for satisfying organizational needs.

The database approach appears to perform well when the data can be structured in terms of elementary fields, and the user knows what he or she wants in terms of those fields. However, when data need to be retrieved from a huge collection of text documents, the traditional database approach does not provide much support. In reality, documents can seldom be structured into database fields.

The purpose of this paper is to analyze the current database approach for retrieving text information and to present an alternative method that uses a hybrid system consisting of a textual database system and knowledge-based system components. The proposed approach is known in the information systems literature as the Intelligent Text Retrieval System(ITRS) or the Intelligent Document Management System(IDMS). The knowledge-based textual database system can significantly reduce the amount of users' effort required to find relevant information. The premise of this research is that important domain knowledge resides already in the large amount of information stored in a database and that a knowledge-based textual database system with intelligent search capability can help users find appropriate information in a timely manner. The research develops a methodology, implements the methodology as a prototype, and finally evaluates the prototype. The specific objectives of the research are:

- *Identification & analysis of commonly occurring problems in textual database systems*
- *Design of a traditional database based system and development of an algorithm to acquire knowledge from the database*
- *Development of a search methodology to use the knowledge base*
- *Design of a prototype ITRS*
- *Comparison of the performances of the ITRS and the database based system*

The rest of the paper has the following structure. Section two explains the general problems of a textual database system. Section three discusses the previous research conducted in the ITRS area. The design of the textual database and the collection of necessary data are explained in section four. Section five illustrates the process of the knowledge acquisition to detect the relationships among articles. Section six discusses the search algorithm employed by the ITRS. Section seven evaluates the performance of the prototype ITRS. Finally, section eight explores the implication of this research.

II. Limitations of Textual Database Systems

Many textual databases exist. Universities such as BGSU have access to BG Link for books and ProQuest for journals. The data retrieval method used by both databases is based on the traditional key-word search techniques such as Boolean for full text and string-matching for index-based. The research described in this paper focuses on the design of a prototype ITRS for finding **journal articles**. In the database retrieval systems, the users type exact key words to find the journal article they want. The system retrieves the articles that contain the key words. It means that users must type the exact key words to find the journals they

want. The purpose of these systems is to provide reference information according to users' request.

If users know the exact author, article name, or keywords, there will be no problem finding what they want. The problem lies in situations where users have ambiguous knowledge about the subjects - for example, assume a user wants to find the journal articles related to the subject, "The Motivational Factors of Early Retirement." If the user types these words, ProQuest cannot find any articles. It is due to the mismatch between the searcher's terms and the index terms, which is referred to as the *term matching* problem[1]. If the user types the key words "early retirement," ProQuest will provide more than 400 related journal articles. It will take enormous time for the user to look at each article's abstract to identify its relevancy to the topic. It is assumed that the abstract is a proper representation of the article's content. Since it is a general rule that "the amount of time spent to retrieve relevant information is proportional to the amount of information stored[2]," the user's effort will be much greater if the number of journal articles in the database increases substantially. The conventional keyword-driven search techniques are inefficient and ineffective in terms of time spent and the relevancy of information found.

Basically, ProQuest cannot determine how closely the journal articles are related to the topic. It is possible that some articles are closely related to the topic but others are not. The strength of the linkage between the information the articles contain and the search topic can vary significantly. An article can be entirely dedicated to the subject or can be partially related to the subject. The user may spend unnecessary time reviewing the articles that have a very low degree of relevancy. Furthermore, the system cannot decide what subject area the user is truly looking for. The result of the psychological reasons of the early retirement will be greatly different from that of the economical reasons of the early

retirement. ProQuest will simply list every journal article related to the topic regardless of the degree of information relevancy. The consequence will be findings that are irrelevant to the user's interest.

It is generally known that the traditional keyword-driven search technique used by database systems miss more than half of the relevant information[5]. Although the user may review all the journals found by the database, he or she is not sure that every article related to the topic in the database has been found. There is still a high possibility that some other important, relevant articles have been missed during the search because of the mismatch between the key words in the user's perception and those contained in the database. The search terms the user uses may not be a true representation of what the user is really looking for. This is the problem of *query articulation*[1]. Therefore, pure database systems are inefficient because users are required to spend a very long time reviewing journal articles, and ineffective because the large portion of findings may not be useful. In addition, there may be other relevant journal articles undetected. Our proposed approach has the potential to assist users finding relevant journals more efficiently and effectively by reducing the term matching and the query articulation problems.

III. Previous Research

There are two approaches to design the IDMS. The IDMS is a knowledge-based system that helps users retrieve information effectively and efficiently. IDMS can be designed by a manual or an automatic knowledge acquisition method. The manual approach is the conventional way of building a knowledge base using knowledge engineers. The librarians are assumed to possess necessary expertise to respond to users' requests. Thus, the necessary knowledge is acquired through interaction with human experts by knowledge engineers. Techniques such

as interviews and observation can be utilized for this kind of knowledge acquisition.

On the contrary, the automatic knowledge acquisition approach is not dependent on human experts for knowledge acquisition. A mathematical algorithm is adopted to extract the knowledge automatically from a textual database. Generally, the knowledge acquisition requires the most time in the development of an IDMS. An automatic knowledge acquisition method has the potential to reduce the time spent in this phase of the development.

3.1 *Manual Knowledge Acquisition Approach*

Peretz Shoval designed an IDMS performing the role of an information specialist using the conventional approach. His objective is to "translate the knowledge and the information processing taking place by the specialist into an intelligent computer program that can carry out the information specialist's task[7]." He suggested a conceptual framework to design the expert system based on the work principles of information specialists. There are three distinct characteristics of the system: interaction between the system and a user, searching for relevant terms, and evaluation of findings.

The system is based on the role of a consultant; "the system accepts the user's problem, which is expressed in user's terminology, and suggests a set of the appropriate index terms to represent the problem[7]." If the user regards a suggested term as irrelevant, the system will go back to the point where the mismatch of terms occurs and will suggest alternative answers. Imitating a human consultant, the system employs an ability to retrace the rejected terms by the user and to find alternatives. Therefore, the interaction between the user and the system is crucial to find the right answers. However, depending on the situation,

the degree of the interaction can vary, because the user's involvement with the system is less required in some events.

The search strategy used to find relevant terms is *the expand and match.* The expand refers to "the process of exciting quiescent knowledge and converting it into active knowledge which is relevant to the problem at hand[7]." The search is activated by expanding a node in a semantic network along its directed links. The expand process is complemented by the match process which tries to find intersections of nodes.

Shoval suggested two criteria to evaluate the terms found by the system at the end of the search process: *the metric of strength* and *complementation.* The strength refers to the number of originators or user terms and the complementation refers to the assembling of a cluster of terms, which are out of the active-relevant terms, to encompass together the meaning of all the user terms. If the set of terms found includes more user terms, they will have higher strength to indicate closer representation of the meaning of the problem. The complementation is utilized to assure that the suggested terms present the user's problem as a whole.

Although there is no direct connection with Shoval's research, there is an interesting project that employed Shoval's idea. Using the conventional approach to acquire the knowledge by the knowledge engineers, Chemical Warfare/Chemical Biological Defense Information Analysis Center(CBIAC) conducted a practical project to build an intelligent document management system. CBIAC faced an information overload problem. Its on-line database had to manage more than 14,000 documents with more than 23,000 citations. It adopted the Intelligent Text Management System(ITMS), a product of Information Access Systems Inc.(IAC), to design a ITRS. The ITMS has a knowledge base, called Judgment Base Module(JBM), that simulates a human judgment technique. The ITMS can handle conventional full text indexing search as well as intelligent

retrieval. After constructing a prototype, CBIAC concluded that the intelligent retrieval capability of the new system was better than the traditional key-word systems[5].

3.2 *Automatic Knowledge Acquisition Approach*

The researchers from the University of Arizona are pioneers in developing the automatic thesaurus generation approach. Most of the research about the ITRS reviewed for my paper is conducted by Hsinchyn Chen and his colleagues at the University of Arizona. Chen and his colleagues have been developing the Intelligent Textual Retrieval System since 1991. They started from the conceptual design to the construction of the complete system that is not a prototype or model but a real life system.

In the early research, Chen and his colleagues explored the human cognitive process used to retrieve relevant terms in a computerized document-based information system which has three components: a database of documents, a classification scheme to index the documents, and an on-line system to access the documents. They employed four data collection techniques such as think-loud protocols, tape recordings of the interactions, interviews, and questionnaires to identify the cognitive process. The subject on-line system was the New York University(NYU) on-line catalog system, called Bobcat, and the participants were the librarians, NYU students ranging from freshmen to Ph.D. candidates, and other users.

They used the *Problem Behavior Graph* (PBG) to explain the on-line search process as the problem solving process. The PBG describes problem-solving activities in a time sequence from an initial state to a goal state. It is composed of semantic elements; the knowledge elements and the operator elements. The knowledge elements are the background knowledge to confront

problems the user has. The operator elements mean a set of actions that obtains an initial state of knowledge as input and produces a new state of knowledge as output. Thus, the terms the user uses in the search are the knowledge elements. The operator elements consist of two classes of operators: *searching* and *scanning*. The searching operator is the core part of the search strategy and the scanning operator examines the summarized or more detailed information about a book. They identified five major search strategies: the *KNOWN-ITEM-INSTANTIATION*, the *SEARCH-OPTION-HEURISTICS*, the *THESAURUS-BROWSING*, the *SCREEN-BROWSING*, and the *TRIAL-AND ERROR*.

The KNOWN-ITEM-INSTANTIATION strategy is conducted when the user possesses some familiarity with the subject area he or she is searching for. In the case that the user knows the exact name of the author and wants to find out the research relevant to the certain subject area, this strategy is mostly utilized. Because of the prior knowledge the user has, there is less possibility of the term matching problem to occur.

The SEARCH-OPTION-HEURISTICS strategy is usually used by the reference librarians and the searchers with good system knowledge. The user who has extensive experience of manipulating the system tends to develop his or her own search heuristics to find relevant terms.

The THESAURUS-BROWSING strategy is designed to alleviate the term matching problem. It is used by many reference librarians to assist searchers. The searcher provides search terms that are usually his or her own terms. The librarian initiates a term translation process to translate the searcher's supplied terms into index terms. Consulting thesaurus with a brain storming process, the librarian finds matching terms.

The SCREEN-BROWSING strategy is a method to browse the list of index terms alphabetically adjacent to the user's search terms displayed on the screen.

The TRIAL-AND-ERROR strategy is an arbitrary search process that the user uses whatever terms in his or her mind to find relevant terms.

They used two statistics-based algorithms, Cosine and Cluster, to extract knowledge automatically from textual data stored in the textual database created by Mosaic international computing research group at the University of Arizona. The application of the algorithms resulted in two knowledge bases that had different structures. The Cosine knowledge base contains symmetric and equally-weighted links. On the contrary, the links in the Cluster knowledge base are asymmetric. Asymmetric links mean that there may be links from A to B, but not from B to A. For example, people can associate *volleyball* and then *net*, but it is hard to associate net and then volleyball. The extracted knowledge was represented in the form of a frame-based semantic network. The semantic network is composed of nodes and links. The node is keywords or index terms and the link is the relationship between the terms. There are four types of the links: narrower terms(NT), broader terms(BT), related terms(RT), and synonymous official terms(USE).

The search algorithms they used were "a heuristics-based spreading activation process for an intelligent on-line catalog" and a branch-and-bound algorithm for guiding the search to "identify the most relevant concept in a large network of knowledge[2]." It calculates a cost, which is a metric to indicate the semantic distance of terms, and tries to find a least-cost path. The branch-and-bound algorithm was designed to avoid a computational explosion problem that occasionally occurs in the semantic network based knowledge-based system. Using recall and recognition tests, they evaluated the performance of the two knowledge bases. The result was that the Cluster knowledge base was better than the Cosine knowledge base in terms of term consistency with the subjects.

Overall, they found that 70% of the terms the knowledge bases suggested were regarded relevant by the subject experts[1] [3] [4].

Another interesting research that uses the automatic knowledge acquisition approach to design IDMS was the one conducted by Salton and his colleagues. They use the vector processing model to represent information retrieval operations. Document vectors are used for comparing index terms and for identifying the relevance between them. Query vectors are used for "representing index items and retrieving user terms found to be similar to the queries[6]." They developed an algorithm to compute the similarity between a query vector and the stored document vector and found term weights indicating the strength of the relevance. The term weight of one is used for assigned terms and zero for missing terms.

One interesting feature of this system is a hypertext that has a network text structure. The nodes of the network represent text excerpts and the links represent various relationships between text segments. The hypertext can be automatically constructed using the vectors. They suggested two kinds of structured text presentation methods to form hypertexts such as mapping and clustering. The mapping is a way to create a web of the texts according to the relationships. In the clustering method, the similar text items were grouped into affinity classes or clusters. Dependent on the method, a different search strategy was used. For the mapping method, a breadth m-depth n search strategy is used and for the clustering, a cluster-based search strategy.

Salton's research suggests an alternative way of the knowledge acquisition. Since the keywords are provided by the authors in the journals such as *the Journal of ACM* and *ACM Transactions on Database Systems*, it is not an unrealistic assumption that they represent the content of the articles. However, in reality, most of articles do not contain all relevant keywords. In the case of journals in which the authors do not provide keywords, this assumption cannot be sustained.

Salton uses a sentence comparison method to find the relevancy between articles. It actually compares entire sentences in articles to determine the strength of the relevancy. Although it has some limitation, the automation of the sentence comparison can significantly reduce the human activity to go through thousands of articles just to find the relevancy.

IV. Design of a Prototype Textual Database

We designed a prototype textual database system for two reasons: a fair evaluation and knowledge acquisition. Since the proposed knowledge-based system is a prototype that contains limited data, it is not logical to compare it with a real life system such as ProQuest. Therefore, the prototype textual database system, which uses the same data as the knowledge-based system, was developed for fair comparison. The database is also a foundation for the knowledge base. As the database is assumed to contain the domain knowledge, the knowledge needed to construct the knowledge base will be extracted from the database. Therefore, the construction of the prototype textual database is an essential process to build the knowledge base.

The data came from two journals published in the last four years. Two hundred articles from *the Journal of Association for Computing Machinery(ACM)* and *the ACM Transactions on Database Systems* were collected for constructing the database. The authors' name, title, journal name, and key words were the main items collected from each article. The keywords that are assumed to represent the content of the articles are provided by the authors.

The textual database includes two tables that are named as *main* and *keyword*. The table main includes four attributes: author, call number, article, and journal. The call number is assigned arbitrarily and the article refers to the title of an article. The keyword includes two attributes: call number and keyword. The

call number is an index key. RBASE® was used for the design of the relational prototype textual database.

V. Knowledge Acquisition from Database

Dependent upon the search strategy, the process of knowledge acquisition can be manual or automatic. For this research, an automatic thesaurus generation approach was adopted to create the knowledge base. The necessary knowledge was automatically acquired by the knowledge acquisition algorithm analyzing the keywords that are part of the journal articles - it is already assumed that the key words represent the content of articles adequately. For the automatic thesaurus generation approach, the two algorithms, Cosine and Cluster, were available. The Cosine algorithm is based on normalized cosine computation which is generally used in automatic text processing[2]. The Cluster algorithm developed by Chen and his colleagues is based on the probability of term co-occurrence in document sets. We adopted the Cluster algorithm for this research, because it is a more rational assumption that the links between terms are asymmetric rather than symmetric.

The Cluster algorithm is designed to calculate the strength of relevance between keywords. The major premise of this algorithm is that the terms will have higher relevancy if they appear together in more document sets. Thus, higher co-occurence of the index terms in the document sets means closer relationship. The Cluster algorithm can be described by the following equations;

$$Weight(T_j, T_k) = \frac{\sum_{i=1}^{n} d_{ij} \times d_{jk}}{\sum_{i=1}^{n} d_{ij}}$$

$$Weight(T_k, T_j) = \frac{\sum_{i=1}^{n} d_{ij} \times d_{ik}}{\sum_{i=1}^{n} d_{ik}}$$

where d_{ij} is a descriptor of keyword T_j in document *i* and d_{ik} is a descriptor of keyword T_k in document *i*. The d_{ij} or d_{ik} just indicates the presence of T_j or T_k in article i and values zero in case of absence or one in case of presence. The denominator indicates how many times keyword T_j appears from first to nth article. The numerator indicates how many articles keyword T_j and T_k appear together. Therefore, if T_j appears in 15 articles and T_j and T_k appear together in four articles, the weight of T_j to T_k will be 26.67 percent. If T_k appears in five articles, the weight of T_k to T_j will become 80 percent. The equations are designed to accommodate the premise.

Based on the Cluster algorithm, we designed a program for knowledge acquisition using Turbo Pascal®. The program used the *keyword* table from the textual database as an input file and produced an output file, "KA.PAS." For programming convenience, we coded the keywords in the *keyword* table and created a new data file, "keyword.pas," for input. The program has two procedures; one for reading the input file and the other for finding the term co-occurence. Since the total number of the keywords in the 200 articles is one thousand and four hundred eighty six, we established a 1,486 by 1,486 matrix to calculate the term co-occurence. The cutoff rate, 0.5%, was adopted to exclude the weight that was regarded as an insignificant relationship[3]. Using a personal computer with a DOS operating system, it was impossible to construct a huge multiple array like the 1,486 by 1,486 matrix due to the limited capability of DOS. Thus, we separately ran four programs that contained one-fourth of the array and then reassembled four output files to one. The result was astonishing. Approximately eleven thousand and three hundred linkages; among keywords, linkage connects two keywords and includes the strength of the link.

VI. Development of A Search Algorithm

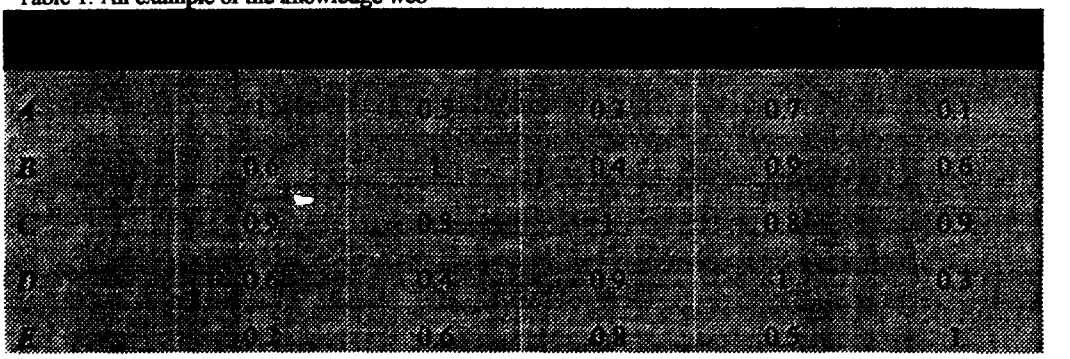
The knowledge acquisition provided the linkages that are in the form of a web. Each keyword is connected to other keywords by link weights showing the strength of their relationship. Table 1 shows an example of the knowledge web. Based on the web as a knowledge representation scheme, a search algorithm was designed by Dr. Raghunathan to find keywords related to a given set of user terms as well as a given minimum link weight. The suggested search algorithm is

1. Obtain a keyword or a set of keywords and a cutoff rate from a user.
2. Obtain all other keywords(or nodes) related to the keyword or a set of the keywords given in step 1 whose link weights are greater than or equal to the cutoff rate.
    2.1 Find new keywords related to the each keyword found or given in step 1 whose link weights are greater than or equal to the cutoff rate.
    2.2 If there are no new related nodes with links higher than the cutoff rate, stop the search.
    2.3 Continue the procedure 2.1 and 2.2 until there are no new keywords found.
3. Calculate the weight of the each node's link to the initial set of keywords using the formula which is the multiplication of link weights residing in between the root keyword and the designated keyword. If the calculated weight of the link from the root node is smaller than the cutoff rate, discard it, else, store it.

Let's assume that the user provides the keyword 'A' and the 0.5 cutoff rate. The keywords connected to the A with the term weights equal to or greater than 0.5 are *B, D*(see table 1). According to the row order of the table or the matrix, the search goes to B. The keyword connected to B whose the strength of the linkage is greater than or equal to the cutoff rate is E. The weight between B and E is 0.6. Since there are no more keywords connected to B, the search goes to D and finds C whose link to D is 0.9. Then, the search goes to C and tries to find a new link. Since the search cannot find a new link, it goes to E. There are no new links connected to E found by the search . If the search did not find any new links for the third level, it stops. Finally, the link weights of the new keywords and A are calculated. The recalculated weight of the link A-E is the multiplication of A-

B and B-E link weights. Thus, the weight of the link between A and E is 0.3(= 0.5 × 0.6). The new weight of the link A-C is the link weight of the D multiplied by that of the C that is 0.63. Therefore, E will be discarded because its link weight to the A is smaller than the cutoff rate. In this example, the related keywords found are A, B, D, and C.

Table 1. An example of the knowledge web



We wrote a Pascal program(see Appendix B) that uses the suggested search algorithm. The characteristics of the implemented search algorithm are *iterative deepening breadth-first* and *heuristics* based. The direction of the search is from the keyword, A, which the user provides, toward the relevant keywords, B, D, and C. Unlike depth-first search which goes deeper into the search space as much as possible, our algorithm explores the space in a level-by-level fashion. The search does not pursue all of relevant keywords, but discriminates against the one that has a link weight below the cutoff rate.

The program uses the file, "knw.txt," created by the knowledge acquisition algorithm as input and generates an output file called as "result.pas." Because of the limitation of DOS operating system, it is not feasible to assign the data from the input file into a multiple array such as 11,000 by 11,000; Turbo Pascal® allows only 64K bytes for declaring variables. Because the input file is not an

index file but a text file, it is not possible to develop a single procedure to finish the search, either. Therefore, *sequential access* to an external file was developed to cope with the generic shortcoming of the personal computer(PC) operating system. In order to avoid using a huge multiple array, the sequential access to an external file was adopted to expand the level of search. The sequential access to an external file refers to the process of reading an external data file whenever the search expands a new level. If it is possible to store the data from the external file into a multiple array, the sequential access will not be necessary. Memorizing each link in the memory by the assignment of each link to a cell of the array , the program can easily retrieve the links from the array and expand any levels of search. Under our circumstances, the search can expand only one level by reading the data file once. It can find the keywords connected to the A, but cannot find the keywords connected to the B or D. To find the E and the C, the data file must be read one more time. To expand three levels of search, the data file must be read thrice. Therefore, the data file will be read whenever the search expands to a new level for finding new links. Through the sequential access to the external file, the program can overcome the limitation of the PC operating system.

The program includes ten procedures to accommodate ten levels of search and one procedure to recalculate the links to the root keyword. There are two arrays to store output data. The first, called *Knowledge*, is for storing the links and the weights and the other, called *Metakmw*, is for storing keywords. The first one stores the findings of the search. It is designed not only storing the output but also the potential possibility of expansion of the system's ability. An intelligent system is different from a generic database in terms of its inherent ability to explain how it reaches the goal state.. Storing the paths, the system can respond to a user's request to explain how to get certain keywords. If the user wants to know how C is related to A, the program can conduct backward search and explain the

connections between C and D and between D and A. The Metaknw array contains every keyword found during the search. It stores every keyword found during the search in the sequential order of finding. After the first level of the search in the example, the links between A and B and between A and D were found. At the second level, there are possible two links B-A and B-E for the keyword, B. Since the link between A and B is already found, the B-A link will not be considered. By storing the keywords A and B in the Metaknw, the program can intelligently discriminate against the relationship that is already found.

The program has limited capabilities to store the data in two arrays and to pursue a level of search at a time. It can store maximum of 900 links and 500 meta-knowledge keywords and conduct eleven levels of search. Due to the limitation of the DOS operating system, it is not possible to create more than 1,400 cells of the arrays. Although the limitations seem to affect the effectiveness of the search, in most of cases, the search may not even expand to beyond the fourth level, because of *the distance effect*. The distance effect is the result of recalculating a link to a root keyword which may dilute the strength of the link. As the distance between a root keyword and other keywords grows, the link weights of the root keyword to the others become significantly affected by the strength of links that reside between them. For example, let's assume that one path to the root keyword has four links(A-B, B-C, C-D, and D-E) and each link has a weight of 0.8, which is considered to be very high. If each link is readjusted to the root keyword, A, at 0.5 cutoff rate, the results will be 0.80 for the A-B, 0.64 for the A-C, 0.51 for the A-D, and 0.41 for the A-E. The strength of the A-E link will be significantly diluted and be discarded by the search. Even with the extremely high link and the moderate cutoff rate, the search does not include the finding of the fourth level. The distance effect will become more profound when the levels

of search expand deeper. Therefore, the limited capacity of the search does not appear to affect the effectiveness.

VII. Evaluation of the KBS

To judge the effectiveness of the prototype knowledge-based system, its performance was compared with that of the prototype textual database. Using the same input, the two systems generated different outputs. By comparing the outputs, it can be determined which system is closer to retrieving the right answers. For accurate evaluation, test queries as input were obtained from a faculty in BGSU who has an in-depth knowledge about the MIS areas. He also reviewed the answers of each query and determined whether they were relevant and whether there were any missing articles. The following ten queries were used:

1. Find articles related to the shortest path algorithms in common networks.
2. Find articles related to discrete mathematics application in information theory.
3. Find articles related to distributed Artificial Intelligence.
4. Find articles related to search algorithms.
5. Find articles related to deductive databases.
6. Find articles related to relational databases.
7. Find articles related to normalization methods in database design.
8. Find articles related to query processing.
9. Find articles related to logic programs.
10. Find articles related to distributed databases.

The queries are designed to evaluate the performance of the textual database and the proposed system in terms of effectiveness.

The retrieved answers from both systems were compared using two criteria, the number of relevant articles retrieved and the number of articles missed. After identifying the number of irrelevant articles and the number of articles missed, we

calculated the percentage of missing and irrelevant articles. The following equations were used to find the irrelevancy rate and the missing rate:

% of missing articles = # of missing articles / # of relevant articles

% of irrelevant articles = # of irrelevant articles / # of articles found.

The system that has a lower missing rate and a lower irrelevant rate is considered to be more effective.

For the database, all the articles that contain the given keywords were retrieved. For the KB system, all of the related keywords were identified at a given cutoff rate and then all the articles that contained all the keywords(AND) as well as at least one of the keywords(OR) were retrieved. The various cutoff rates were used to analyze its effect on retrieval. The total number of relevant articles for each query comes from the search that generates the maximum number of relevant articles.

The output from the first query shows that the database outperforms the KB. The database system found three articles - two of them are relevant - and the KB system found four relevant articles. In the OR search, the KB system found a total of 32 articles at 0.8, 0.9, and 1.0 cutoff rates and 65 at the other cutoff rates; there are four relevant articles at any cutoff rate. In the AND search, the KB system found a total of three articles with the two relevant at the 0.8, 0.9, and 1.0 and one article, which is irrelevant, at the other rates. The OR search of the KB system did not miss any relevant articles but has very high irrelevancy rates at all the cutoff rates. The high irrelevancy rate implies that the retrieved information is not really useful in the case of the novice user who does not possess an in-depth subject knowledge, so he or she cannot distinguish a relevant article from an

Table 2. Query #1

		irrelevancy rate	missing rate
Database		0	0.500
Knowledge base	cutoff rate		
And	>= 0.5	1.000	1.000
	0.6	1.000	1.000
	0.7	1.000	1.000
	0.8	0.333	0.500
	0.9	0.333	0.500
	1.0	0.333	0.500
Or	>= 0.5	0.938	0
	0.6	0.938	0
	0.7	0.938	0
	0.8	1	0
	0.9	1	0
	1.0	1	0

irrelevant article. In the case of the database, the irrelevant rate is zero but the missing rate is 50 percent. The high missing rate means that there may be important articles undetected during the browsing. The database approach is clearly better than the AND KB approach.

In the second query, the comparison is inconclusive. The database found one article that is relevant. The KB found 33 articles at 0.8, 0.9, and 1.0 cutoff rates and 60 articles at the other rates during the OR search. At all the rates, it found three relevant articles. During the AND search, the KB found one relevant article at 0.9, 0.9, and 1.0 and nothing at the other rates. The comparison between the OR KB and the database is inconclusive. Dependent on the evaluator's

Table 3. Query #2

		irrelevancy rate	missing rate
Database		0	0.667
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	n/a	1
	0.7	n/a	1
	0.8	0	0.667
	0.9	0	0.667
	1.0	0	0.667
Or	>= 0.5	0.950	0
	0.6	0.950	0
	0.7	0.950	0
	0.8	0.909	0
	0.9	0.909	0
	1.0	0.909	0

preference on the rates, the KB or the database approach may be better. In the AND search, the KB did not find any article at 0.5, 0.6, and 0.7 cutoff rates. The

database shows an advantage over the KB at those cutoff rates. At the cutoff rate ranging from 0.8 to 1.0, the performance of the KB is same as that of the database.

In the third query, it is inconclusive to make a judgment on which system is better. The database found one article that is relevant. The KB found a total of 65 articles with three relevant at the cutoff rate ranging from 0.7 to 1.0 and a total of 149 articles with three relevant at the other rates during the OR search. During

Table 4. Query #3

		Irrelevancy rate	missing rate
Database		0	0.667
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	n/a	1
	0.7	0	0.667
	0.8	0	0.667
	0.9	0	0.667
	1.0	0	0.667
Or	>= 0.5	0.980	0
	0.6	0.980	0
	0.7	0.954	0
	0.8	0.954	0
	0.9	0.954	0
	1.0	0.954	0

the AND search, the KB found one article, which is relevant, at 0.7, 0.8, 0.9, and 1.0 cutoff rate and nothing at the other rates. It is inconclusive to determine which system is better when the OR KB approach is compared with the database approach. The AND search at the cutoff rates ranging from 0.7 to 1.0 produced the same output as the database. However, the AND search at 0.5 and 0.6 cutoff rates did not find any articles. The database approach is clearly better over the AND KB approach at 0.5 and 0.6 cutoff rates.

In the fourth query, the KB approach outperforms the database approach in the OR search. The database found one irrelevant article. The KB found a total of 78 articles at 0.5 cutoff rate in the OR search. Among them, six are relevant. In

Table 5. Query #4

		irrelevancy rate	missing rate
Database		1	1.000
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	n/a	1
	0.7	n/a	1
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1
Or	>= 0.5	0.923	0
	0.6	n/a	1
	0.7	n/a	1
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1

both AND and OR, the search did not find anything at the cutoff rates ranging from 0.6 to 1.0. At the 0.5 cutoff rate, the OR KB system resulted in the lower rates than the database system. In the AND KB approach, nothing was found, so there is no performance difference between the two systems.

Table 6. Query #5

		irrelevancy rate	missing rate
Database		0.500	0.800
Knowledge base	cutoff rate		
And	>= 0.5	0	0.800
	0.6	0	0.800
	0.7	0	0.800
	0.8	0	0.800
	0.9	0	0.600
	1.0	0	0.600
Or	>= 0.5	0.952	0
	0.6	0.952	0
	0.7	0.952	0
	0.8	0.952	0
	0.9	0.861	0
	1.0	0.861	0

In the fifth query, the KB seems to perform clearly better than the database. The database found a total of two articles - one of them is relevant. The KB found five relevant articles at all the rates in both searches. In the OR search, the KB found a total of 36 articles at 0.9 and 1.0 cutoff rates and a total of 105 at the other rates. In the AND search, the KB found a total of two articles, which are relevant, at the 0.9 and 1.0 and one relevant article at the other. The KB approach in the AND has the lower missing and irrelevancy rates at the 0.9 and 1.0 and has the lower irrelevancy rate at the other.

Table 7. Query #6

		irrelevancy rate	missing rate
Database		0.200	0.822
Knowledge base	cutoff rate		
And	>= 0.5	0.100	0.800
	0.6	0.100	0.800
	0.7	0.100	0.800
	0.8	0.100	0.800
	0.9	0.100	0.800
	1.0	0.100	0.800
Or	>= 0.5	0.563	0
	0.6	0.563	0
	0.7	0.563	0
	0.8	0.563	0
	0.9	0.563	0
	1.0	0.563	0

In the sixth query, the performance of the KB is better in the AND and very competitive in the OR. The database found a total of ten articles; two of them are irrelevant. The KB found a total of 103 articles at all the cutoff rates in the OR. Among them, 45 are relevant. In the AND search, the KB found ten articles; only one is irrelevant. In the OR search, the KB did not miss any relevant articles and has a relatively low irrelevancy rate. Although the KB has the higher irrelevancy rate than the database, almost half of the findings are relevant. The moderate irrelevancy rate makes the KB competitive.

Table 8. Query #7

		irrelevancy rate	missing rate
Database		0	0.750
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	n/a	1
	0.7	n/a	1
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1
Or	>= 0.5	0.976	0
	0.6	n/a	1
	0.7	n/a	1
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1

In the seventh query, the database has better performance. The database found a total of one article that is relevant. When the cutoff rate ranged from 0.6 to 1.0, both searches found no articles. At 0.5 cutoff rate, the KB found a total of

166 articles; four of them are relevant in the OR and nothing in the AND. The comparison between the database and the OR search at the 0.5 cutoff rate is inconclusive, but the rest of the KB's performance is worse than the performance of the database.

In the eighth query, the database has better performance over the AND KB. The database found a total of four articles; one of them is not relevant. The KB found a total of 84 articles - 20 are relevant - at 0.8, 0.9, and 1.0 cutoff rates, a total of 156 articles - 28 are relevant - in the OR search, and a total of 157 articles at 0.5 rate. In the AND search, the KB found a total of four articles - two of them are relevant - at 0.8, 0.9, and 1.0 cutoff rates, total three articles - all of

Table 9. Query #8

		Irrelevancy rate	missing rate
Database		0.250	0.893
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	0.333	0.929
	0.7	0.333	0.929
	0.8	0.500	0.929
	0.9	0.500	0.929
	1.0	0.500	0.929
Or	>= 0.5	0.822	0
	0.6	0.821	0
	0.7	0.821	0
	0.8	0.773	0.286
	0.9	0.773	0.286
	1.0	0.773	0.286

them are relevant - at 0.6 and 0.7 rates, and nothing at the 0.5 rate. The comparison between the database and the OR KB is inconclusive. However, the database has the lower missing and irrelevancy rates than the KB in the case of the AND.

In the ninth query, the database shows better performance compared with the AND KB. The comparison between the database and the OR KB is inconclusive. The database found one article that is relevant. At 0.8, 0.9, and 1.0 cutoff rates, both searches did not find any articles. The KB found a total of 123

articles - nine of them are relevant - in the OR search and nothing in the AND search at 0.5, 0.6, and 0.7 cutoff rates.

Table 10. Query #9

		Irrelevancy rate	missing rate
Database		0	0.889
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	n/a	1
	0.7	n/a	1
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1
Or	>= 0.5	0.927	0
	0.6	0.927	0
	0.7	0.927	0
	0.8	n/a	1
	0.9	n/a	1
	1.0	n/a	1

In the tenth query, it seems that the performance of the KB ties with that of the database. In the case of the OR search, the evaluation is inconclusive. Except at a 0.5 cutoff rate, the KB has the same missing and relevancy rates. The database found a total of one article that is relevant. In the OR search, the KB found a total of 128 articles - 20 of them are relevant - at a 0.5 cutoff rate and a total of 125 articles - 21 of them are relevant - at the other rates. In the AND search, the KB found nothing at a 0.5 cutoff rate and one at the other rates.

Table 11. Query #10

		Irrelevancy rate	missing rate
Database		0	0.952
Knowledge base	cutoff rate		
And	>= 0.5	n/a	1
	0.6	0	0.952
	0.7	0	0.952
	0.8	0	0.952
	0.9	0	0.952
	1.0	0	0.952
Or	>= 0.5	0.844	0.048
	0.6	0.832	0
	0.7	0.832	0
	0.8	0.832	0
	0.9	0.832	0
	1.0	0.832	0

In all the queries, the evaluation between the database and the OR KB is almost inconclusive. Only in the fourth query, the OR KB outperforms the

database. Therefore, the comparison between the database and the AND KB becomes a core element to determine which system is better. Besides the database approach is very similar to the AND KBS approach - they retrieved all the articles containing all the keywords. It may be more logical to make a judgment based on the comparison between the database and the AND KB. The AND KB shows better performance than the database in the three queries. The database performs better than the AND KB in the four queries. The comparison between the AND KB and the database is inconclusive in two queries and resulted in the tie in one query. Overall, the performance of the database is better than that of the AND KB in the ten queries we represented with.

VIII. Conclusion

The fundamental purpose of human activities with the database system is problem solving. A problem faced by the database management system user is the lack of knowledge related to the subject he or she is interested in. The database system fails to solve the user's problem when data are in the form of the unstructured collection of text documents. In this research, we presented an alternative approach, the knowledge base, to assist the user in finding relevant articles more effectively. The knowledge base can help the users of systems such as the library text database systems browse more effectively. By integrating the knowledge base to the text database, the users will spend less time finding the journal articles that contain relevant information to the subjects they are looking for, so they can spend more time on the rest of research. The system , we believe, will eventually increase the users' productivity.

Using the knowledge-based approach, we designed a prototype Intelligent Text Retrieval System. The design of the knowledge base consists of three major stages: knowledge acquisition, knowledge representation & search algorithm, and

evaluation. For the purposes of the knowledge acquisition and comparison, a prototype database was designed. The Cluster algorithm developed by Chen and his colleagues was adopted to extract necessary knowledge from the database[1]. We used the knowledge web as a knowledge representation scheme and developed the search algorithm. Finally, the performances of the database and the knowledge base were compared.

In spite of the goal to develop a more effective system, the comparison between the KB and the database produced mixed results. The performance of the database is better than that of the KB that used the AND operator in keywords. The comparison between the KB using the OR operator and the database is inconclusive. In the nine queries, the OR KB found all of the relevant articles as well as the many irrelevant articles, whereas the database system found a few relevant articles, but missed the many other articles that are relevant.

The major limitation of this research is the size of the systems we studied. The size of the database in this study is about 200 articles whereas the real life systems may contain millions of articles, so the proposed approach may just ascertain the possibility of improvement, not the guarantee of improvement. The other problems are similar to the generic problems of an expert system. The expert system does not always provide optimal solutions but good solutions in the case of the adaptation of heuristics. The general reasons to employ heuristics for an expert system design are that "a problem may not have an exact solution because of inherent ambiguities in the problem statement or available data and a problem may have an exact solution, but the computational cost of finding it may be prohibitive[1]." An exhaustive search algorithm to review every journal may

[1]Luger, George F. and Stubblefield, William A. Artifical Intelligence and the Design of Exper System, Redwood City: the Benjamin/Cummings Publishing Company, Inc.,1989, pp. 149-150.

require enormous time. Therefore, the proposed system may suggest not optimal solutions but practical solutions.

We conclude that improvements are possible to make the OR KB system more effective. Overall, the OR KB system has a zero missing rate, but a very high irrelevancy rate. The high irrelevancy rate significantly affects the effectiveness of the system. By reducing the high irrelevancy rate, the OR KB can outperform the database. There are several ways to increase its effectiveness. The improvement can be made by changing the database size, finding the strength of the connection between a keyword and an article, implementing an interface with user feedback, and experimenting with different cutoff rates. One of the assumptions of this research is that the important domain knowledge resides in the large amount of information stored in a database. The size of the prototype database may not be large enough to contain the domain knowledge, so an increase in the number of articles stored in the database can help extract more complete knowledge from the database. Another assumption of this research is that keywords represent the content of articles. However, it is not clear whether that assumption is really valid. It is possible that one keyword can be solely dedicated to an article and partially dedicated to other articles. The Cluster algorithm can identify the relationship between keywords, but cannot find the strength of the connection between a keyword and an article. The sentence comparison method developed by Salton can be useful in identifying the strength of the connection between a keyword and an article[6]. If that strength can be identified, the KB can eliminate articles that have low connection strength to the keywords and eventually decrease the irrelevancy rate. The development of a more interactive KB system can also help exclude irrelevant articles effectively. It is possible that the relationship between the keywords discovered by the Cluster algorithm may not be accurate. At each level of the search, the KB system can be

designed to ask a user whether or not the keywords found are related to the subject. By making judgments about the relevance of the keywords found, the user can direct the search to exclude the irrelevant keywords. The cutoff rate can also contribute to reduce the irrelevancy rate. The too high cutoff rate can eliminate some important keywords. The too low cutoff rate can also be a factor in the high irrelevancy rate. The simulation of the search with various cutoff rates can help find an appropriate cutoff rate. These are subject areas to be explored for future study.

References

[1] H. Chen and V. Dhar, "Cognitive Process as A Basis for Intelligent Retrieval Systems Design," *Information Processing & Management*, Vol. 27, No. 5, 1991, pp. 405-432.

[2] H. Chen and K. J. Lynch, "Automatic Construction of Networks of Concepts Characterizing Document Databases," *IEEE Trans. Systems, Man and Cybernetics*, Vol. 22, No. 5, Sept./Oct. 1992, pp. 885-902.

[3] H. Chen, K. J. Lynch, K. Basu, and T. Dorbin Ng, "Generating, Integrating, and Activating Thesauri for Concept-based Document Retrieval," *IEEE Expert*, April 1993, pp. 25-34.

[4] K. J. Lynch and H. Chen, "Knowledge Discovery from Historical Data: An Algorithmic Approach," *IEEE*, Feb. 1992, pp. 70-79.

[5] D. McGonigel and L. J. Golly, "Lost in Space? An Intelligent Retrieval Solution," *AI EXPERT*, June 1993, pp. 50-53.

[6] G. Salton, J. Allan, and C. Buckley, "Automatic Structuring and Retrieval of Large Text Files," *Communication of the ACM*, Vol. 37, No. 2, Feb. 1994, pp. 97-108.

[7] P. Shoval, "Prinicples, Procedures and Rules in An Expert System for Information Retrieval," *Information Processing & Management*, Vol. 21, No. 6, 1985, pp. 475-487.